フラクタルとCG

著者	大関 慎
雑誌名	北海道女子短期大学研究紀要
巻	24
ページ	165-180
発行年	1989
URL	http://id.nii.ac.jp/1136/00001683/

フラクタルとCG

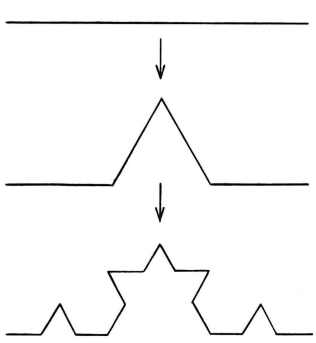
A Note on Fractal and Computer Graphics

大　関　　慎
Shin　　Ozeki

Ⅰ　は　じ　め　に

フラクタルというと，その創始者 B. B. Mandelbrot の名前とコッホ曲線をまず思い浮かべることと思う[1]。我々の前報告によれば，コッホ曲線は，図1のように各辺を再帰的に分割していくことで作成されるものである[2]。コッホ曲線に見られるような「自己相似性」，つまり全体の形状が細部でも繰り返し現れるのがフラクタル図形である。

図1　コッホ曲線



コンピュータ・グラフィックスの大きなテーマの一つに自然現象の表現というものがある。自然現象には，山，海，川，木，空，雲などいろいろあるが，多くのモデル化にフラクタル理論が役立っている。

複雑な形状をした山岳や海岸線などを，正確に表現しようとすると，データ量はかなりの量になってしまう。こういった複雑な自然形状を，少ないデータで近似的に表現するのにフラクタルはよく使われている[3][4]。例えば山を描く場合，大まかな形状を表すデータだけ保持しておき，これらの間の形状はフラクタルで生成させて補間していくと，データ量は極端に少なくなる。

本ノートの目的は，フラクタル理論をコンピュータ・グラフィックスへ利用し，パーソナル・コンピュータにより自然現象を描くことである（自然物の対象は山）。

以下，本報告のⅡ章では，山を描くためのデータ作成に利用した中点変位法，Ⅲ章，Ⅳ章では，3次元処理のためのワイヤーフレーム法，よりリアルに山を表現するためのレイ・トレーシング法について簡単に説明し，コンピュータ・グラフィックスによる自然物表現法を明らか

1）Benoit B. Mandelbrot : The fractal geometry of nature, W. H. Freeman and Company , 1985（広中平祐訳：フラクタル幾何学，日経サイエンス社，1985）

2）大関慎，藤井雅章：フラクタル次元の二，三の考察，北海道女子短期大学，第23号，1988，pp. 261－263

3）宮田一乗：CGや画像解析に応用できるフラクタル，日経CG，NO . 3，1989

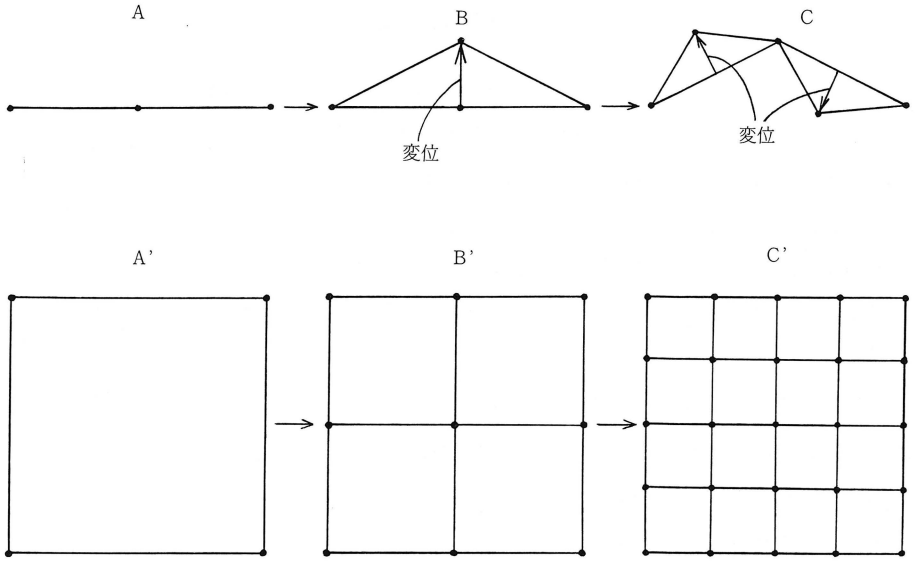
4）安生健一：簡単なパラメータの指定で山，雲，波などを描く拡張フラクタル，日経CG　NO . 9，1988

にする。

Ⅱ 中点変位法

図2のAのように，線分の中点をとり，その中点を上か下に，ある分だけ変位させる。その変位の大きさはランダムに決める。その操作をB，Cのように繰り返し行う。このようにすると，2点を結ぶ折れ線ができあがり，本当の海岸線のようにみることができる。

図2　2次元，3次元の中点変位法



本ノートではこの操作を3次元図形において行わなければならない。3次元で行う場合は，正方形を分割させて変位させる。図2のA'のような正方形を考える。そして，その正方形の各々の辺の中点をとることによって図2のB'のように最初の半分の大きさの正方形がとれる。この操作を繰り返すことによって，次々と細かい正方形ができあがる。最初の正方形から，半分の大きさの正方形ができるわけで，フラクタルの特徴である「自己相似図形」といえる。

つぎに，図2の正方形の頂点の高さを考える。A'の頂点の高さを基準値として，正方形の各中点の高さを図2の2次元と同じ方法で算出する。ここではBの中心の高さは，正方形の4つの頂点の高さの平均値をもとに乱数で計算した[5)6)]。中点変位法によるデータ作成のプログラムは付録のPROGRAM 1に示す。

Ⅲ ワイヤーフレーム法

1．簡単な立体モデル

3次元図形をコンピュータのディスプレイに表示する場合，もっとも簡単な立体モデルとし

5）太田昌孝：三次元フラクタルとCG, Computer Today, NO. 33, 1989, PP. 58–59
6）高橋哲雄：ハロー！フラクタル，OH! PC, NO.7, 1987, PP. 259−262

てワイヤーフレームモデルが利用されている。図3は建築物をワイヤーフレームモデルで表現したものである。[7] 図3が示すように物体の稜線だけで表現する図形で，比較的簡単に複雑な形状を表現できるので，現在もっとも広く用いられている立体モデルである。

ここでは，表現の自由度，データの量を考え，ワイヤーフレームモデルを利用する。

2．ワイヤーフレームモデルの手順[8]

3次元図形をコンピュータなどのスクリーンに表示する場合，座標変換の概念を考えなければならない。座標変換とは3次元図形の平行，回転等の移動後の座標を決めることである。

3次元空間で用いる座標系として視点座標系を用いて表す。視点座標系を用いる理由は，視点が動く場合の座標変換処理の速さである。

たとえば，視点座標系，図4において視点が上を向く場合は，座標の原点，つまり視点を回転の中心として3次元図形を回転すればよい。

視点座標系の座標変換を考える場合，次の3つの考え方がある。

○視点の平行移動→3次元図形が逆の方向へ平行移動したことと同じ

○視点の回転→3次元図形が視点を中心として逆の方向へ回転したことと同じ

○3次元図形自身を中心としての回転→視点が3次元図形を中心として逆の方向へ回転したことと同じ

(1) 平行移動

図5において視点が平行移動して，点T（X，Y，Z）が，点T'（x，y，z）になった場合を考える。X，Y，Z各軸方向の視点移動量を各々

図3　ワイヤーフレームモデルによる建築物の表示例

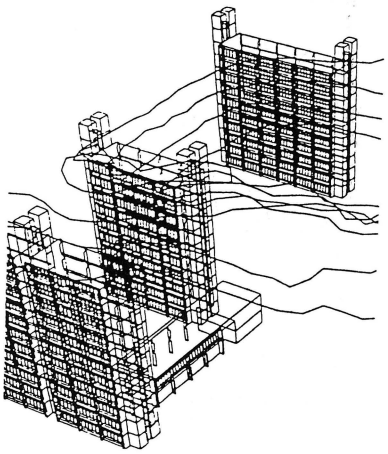


図4　視点座標での視点の回転

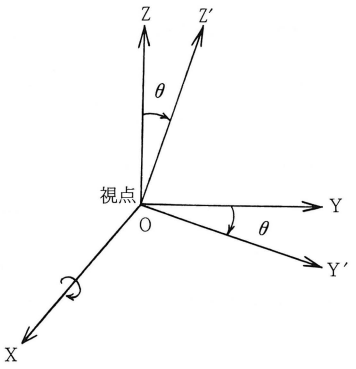
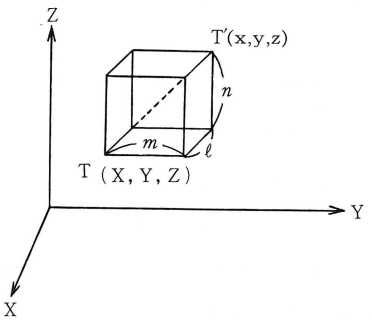


図5　平行移動



7）安居院猛，中嶋正之：コンピュータ・グラフィックスの技法，アスキー，1986，PP.58-59
8）佐藤義雄：入門グラフィックス，アスキー，1986，PP.143-168

l，m，nとすると，移動の関係式は次のようになる。

$$X = x + l$$
$$Y = y + m$$
$$Z = z + n$$

行列表現すると

$$[X \quad Y \quad Z] = [x \quad y \quad z \quad 1] \quad Td$$

変換行列Tdは，

$$Td = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ l & m & n \end{bmatrix}$$

である。

この式を用いて，3次元図形の各頂点に座標変換を行うことによって平行移動が可能になる。

(2) 視点の回転

視点座標系の座標回りの回転は図6に示すように3種類ある。X，Y，Zの各軸回りの回転をBANK, PITCH, HEADING，という。

いま，視点がX軸の回りにθ x回転したとする。このときの座標値の変化は，初期値の座標系における点Pの座標を（x，y，z）とするならば，座標系がθ Xだけ回転して新しい座標値（X，Y，Z）になったことになる。X軸回りの回転による座標変換の関係式は次のようになる。

$$X = -x$$
$$Y = -y\cos\theta x + z\sin\theta x$$
$$Z = -y\sin\theta x - z\cos\theta x$$

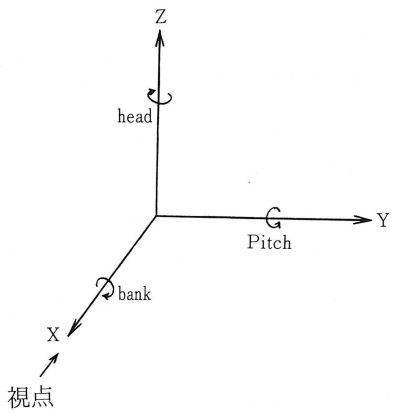
行列表現すると

$$[X \quad Y \quad Z] = [x \quad y \quad z] \quad Trx$$

変換行列Trxは

$$Trx = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -\cos\theta x & -\sin\theta x \\ 0 & \sin\theta x & -\cos\theta x \end{bmatrix}$$

図6 各軸についての回転方向



平行移動のときと同様に，3次元図形の各頂点についてこの座標変換を行えば3次元図形のX軸回りの回転が完了する。

Y軸，Z軸の回転についてもX軸と同様に考えることができる。各々の変換行列は次のようになる。

$$Try = \begin{bmatrix} \cos\theta y & 0 & -\sin\theta y \\ 0 & 1 & 0 \\ -\sin\theta y & 0 & -\cos\theta y \end{bmatrix} \quad Trz = \begin{bmatrix} \cos\theta z & \sin\theta z & 0 \\ \sin\theta z & -\cos\theta z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(3) 変換行列の連結処理

(2)で示した変換行列を組み合わせて（平行移動と回転），3次元図形の座標変換を考えるとき，異種間の変換行列の連結の処理の問題が出てくる。たとえば，平行移動の変換行列と回転移動の変換行列を掛けると行と列の数が一致しないため演算はできない。そこで，変換行列同士の連結を自由に行えるように，変換行列にダミーの行と列を加えて，4行4列として同行同列の行列にしておく必要がある。そうするとこれまでの変換行列は次のように表現できる。

①視点の平行移動

$$[X \quad Y \quad X \quad 1] = [x \quad y \quad z \quad 1] \; Td$$

$$Td = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ l & m & n & 1 \end{bmatrix}$$

②視点のX軸回りの回転

$$[X \quad Y \quad Z \quad 1] = [x \quad y \quad z \quad 1] \; Trx$$

$$Try = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -\cos\theta x & -\sin\theta x & 0 \\ 0 & \sin\theta x & -\cos\theta x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

③視点のY軸回りの回転

$$[X \quad Y \quad Z \quad 1] = [x \quad y \quad z \quad 1] \; Try$$

$$Try = \begin{bmatrix} \cos\theta y & 0 & -\sin\theta y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta y & 0 & -\cos\theta y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

9） CGで図形の座標変換を行うときに用いる手法。3次元座標変換の場合は，4行4列で示される変換行列を用いることにより，一般的に座標変換ができる。主な性質は，

(1)3次元の直線は，変換後も直線である。線分上の点の比は保存される

(2)3次元の平行な直線は，変換後も直線である

(3)3次元の平面は，平面に変換される

である。

④視点のZ回りの回転

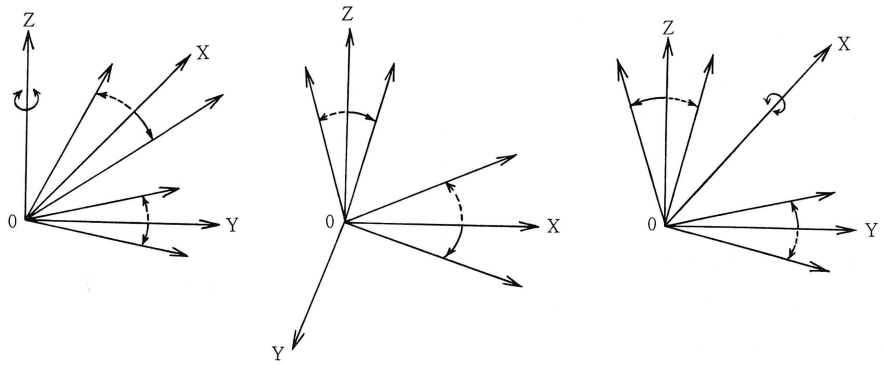
$$[X\ \ Y\ \ Z\ \ 1] = [x\ \ y\ \ z\ \ 1]\ \mathrm{Trz}$$

$$\mathrm{Trz} = \begin{bmatrix} \cos\theta z & \sin\theta z & 0 & 0 \\ \sin\theta z & -\cos\theta z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

このような変換をアファイン変換という[9]。

2つの行列を連結するときにその順序を逆にすると，一般的に結果が異なる。3次元図形の回転の場合も，その順序によって結果が異なるので，その順序に注意しなければならない。通常，3次元空間で物体を見る場合，図7のように，Z軸回りで物体を見つけその大きさを調べる。次に，Y軸回りで物体の高さを調べる。さらにX軸回りを観察する。本ノートで扱う3次元グラフィックスの回転順序もこの動作に対応させて，Z軸→Y軸→X軸で行うこととした。

図7　回転の順序



（a）Z軸回りの回転　　（b）Y軸回りの回転　　（c）X軸回りの回転

変換行列は，各軸回りの変換行列をZ，Y，Xの順に連結すると次式のようになる。

$$\mathrm{Trzyx} = \begin{bmatrix} \cos\theta z\cos\theta y & \sin\theta z & -\cos\theta z\cos\theta y & 0 \\ \sin\theta z\cos\theta y & -\cos\theta z & -\sin\theta z\sin\theta y & 0 \\ -\sin\theta\ 0 & 0 & -\cos\theta y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -\cos\theta x & -\sin\theta x & 0 \\ 0 & \sin\theta x & -\cos\theta x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

代数式は次のようになる。

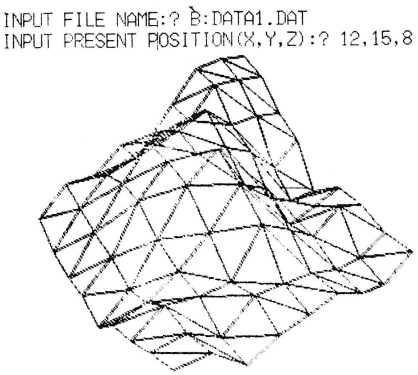
$X = x(\cos\theta z\cos\theta y) + y(-\sin\theta z\cos\theta y) + z(\sin\theta y)$

$Y = x(\sin\theta z\cos\theta x + \cos\theta z\sin\theta y\sin\theta x) + y(\cos\theta z\cos\theta x - \sin\theta z\sin\theta y\sin\theta x) + z(-\cos\theta y\sin\theta x)$

$Z = x(\sin\theta z\sin\theta x - \cos\theta z\sin\theta y\cos\theta x) + y(\cos\theta z\sin\theta x + \sin\theta z\sin\theta y\cos\theta x) + z(\cos\theta y\cos\theta x)$

以上の関係式から，視点を入力することにより，Ⅱ章で述べた中点変位法のデータを利用して実行したワイヤーフレームモデルを図8に示す。（プログラムリストは付録のPROGRAM 2）

図8　ワイヤーフレーム法による山



Ⅳ　レイ・トレーシング法

1．レイ・トレーシング法の原理

レイトレーシング法は，光源から視点に至るまでの光線の軌跡を視点から光源へと逆向きに追跡することで，視点から見ることのできる物体の映像を決定していく方法である。手順は次の3つに分けられる。

(1) スクリーン上の表示領域内にある1つのピクセル[10]を起点に，視線に沿った半直線を定義して，半直線と物体との最初の交点を求める。

(2) その交点における物体の情報と環境の情報（光線の状態や他の物体の影響など）を考慮して，交点での色調を決める。ただし，物体が鏡面反射などする場合には，ここで計算を終了せずに光線の追跡を続け，さらに次の物体との交点を求める。

(3) 決定した色調で起点のピクセルを塗る。

この(1)から(3)を表示領域内の全てのピクセルに対して繰り返す[11]。

rayとは「光線」，traceとは「追跡」と言う意味である。つまり，「光線追跡法」ということになる。しかし，実際には，視線を追跡して，視線に入ってくる光源を調べるアルゴリズムである。

図9で，点Pを視点に二つの立体を見たときの様子を，レイトレーシングで表示することを考えてみる。図9のスクリーン上の点Qの色を計算するにはまず，PからQ方向に視線を延ばす。すると，3角錘と点Rで交わる。さらに点Rで反射した視線は球と点Sで交わる。この先，Sで反射した視線は何にも衝突せずに直進する。

物体はすべて全反射するものではない。いまここで視線を逆にたどってみたが，たどった先に何もなかったとしてもその視線に光りが，通っていないわけではない。

物体を見たときにその物体のある点から出る光には，反射光と乱反射の光がある。いまたどっ

10）グラフィック画面に表示できる最小の領域（ドット）の大きさ

11）豊元詮：レイ・トレーシング法をBASICプログラムで解説，日経CG，NO.11，PP.178-179

図9　レイトレーシング適用例

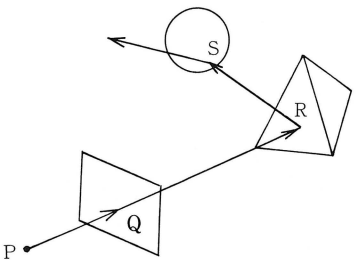
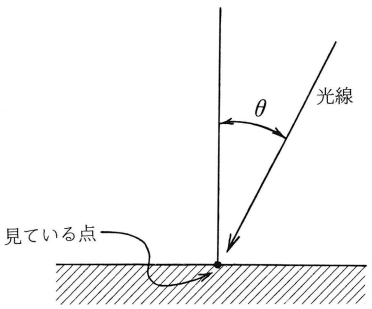


図10　光の反射



たような方法で調べることのできるのは反射光だけである。

乱反射光は物体に衝突したときに，その衝突した点の面が，光源となる光線とどの様な角度をなしているかで決まる。図10のように平面上にある点を見ているときは，この点の明るさは，光線とこの平面の法線（平面と垂直な直線）とのなす角度をθとしたき，光線の強さの$\cos\theta$倍となる。この明るさは，この点をどこから見ていようと乱反射であるので同じであると考えられる。この値を見て分かるように，一番明るいのはθが0のときで，一番暗いのがθが90度のときである。（感覚的に見ても分かる）

乱反射についても注意が必要である。それは，いくら乱反射といえども全ての色がみな一様に反射するとは限らないからである。色によって反射しやすいもの，反射しにくいもの様々である。そこでレイトレーシング法では面に光の3原色（RGB[12]）各々についてどれだけ反射するかを表す反射率を定義しておく。

図9において，点Rで反射するときには乱反射の光が加わる。乱反射の成分は光源からの光との角度を計算すればよいので難しくはない。Rで反射する光を考えると，また球上の点Sで反射する。ここでの様子もRでの反射と同じように考えるとよい。つまり，乱反射を加えることにして，反射光を追跡して行くのである。しかし，こうして反射を追跡しても光源に向かうわけではない。そこで実際のプログラムは何回か反射させたならば，そこで視線の追跡をやめる。つまりS以上追跡しないことにする。Sの乱反射から逆に追跡して行くのである。Sで乱反射した光がRに届き，ここで反射することを考える。

Rにおける反射率をこの光の強さに乗じ，それにRでの乱反射光を加えればできる。実際の光もおそらくこれに近いと考えられる[13)14)]。

2．光源追跡の計算

計算しなければならないことは，次の3つである。

(1)　視点から視線の直線の方程式の計算

12）光の3原色，赤（red），緑（green），青（blue）

13）高橋哲雄：ハロー！フラクタル，OH！PC, NO.8, 1987, PP.285-287

14）Andrew S. classner : Computer Graphics User's Guide, 1984（白田耕作訳：図説コンピュータ・グラフィックス，アスキー，1985, pp.174-182）

(2) (1)物体との交点の計算

(3) (2)の交点における乱反射の光の強さと視線の反射の計算

これらの計算から画面を作成するには，その画面のひとつひとつのドット（画素）について計算をしなければならない。

例えば，画面が320×200の画素からできているとすると，画素は64000個あることになる。これだけの数だけ前に述べた計算をするということは，異常なほど長い時間がかかることが予想さる。本ノートのプログラムでは，画素を正方形として考えているので100×100とした。1つの画素は4×4のドットからできていることになり，画素の中にあるドットの数は16個になる。つまり，この画素1つにつき，1つの色は0から16の17段階の強さで表示することができることになる。コンピュータ・ディスプレイは3原色を用いているので，

$$17\times17\times17=4913$$

の色を表現できることになる。この画素のことを「ピクセル」と呼ぶ。

このピクセルの中では，その17段階の明るさについて，どのドットが表示されるか決められている。1つのピクセルの中ではできるだけ固まらないように，表示領域を散らさなければならない。図11のようにピクセル内のドットに番号を付けて，0から16の段階の明るさについて点灯させるようにする。例えば6という明るさならば6以下の数字の付いているドットを点灯させるという意味である。

本ノートのプログラムでは各々の色について256が上限値としている。値は整数でも実数の値でもよい。

図12のAのように256を16等分する。例えば，明るさが16とか32のような数になった場合は，ピクセル中のドットを1個あるいは2個点灯させればよい。しかし，中途半端な場合，例えば明るさが8だったとする。図12のBのように拡大して考えると，8は0と16のちょうど中間の値であるので，1つ点灯させる確率とまったく点灯させない確率を0.5とする。つまり，明るさが8のピクセルを数多く表示すると，そのうちの半分はドット1つが点灯し，残りの半分は全く点灯しないことになる。どのピクセルで点灯させないかはランダムに決めることとする。

図11　ピクセル内の点灯方法

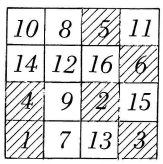
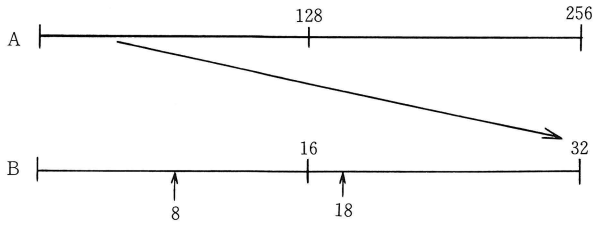


図12　ランダム・デイザ法



15) 太田昌孝，竹内あきら，大口孝之：応用グラフィックス，アスキー，1986

16) 高橋哲雄：ハロー！フラクタル，OH！ PC, NO.8, 1987

このように乱数を用いて，ピクセルにより点灯するドット数を変える方法を「ランダム・ディザ法」という[15)16)]。

以上の関係式から，視点を入力することによりⅡ章で述べた中点変位法のデータを利用して実行したレイ・トレーシングモデルを図13に示す。（プログラムリストは付録のPROGRAM 3）

Ⅱ章で作成した中点変位法と同じデータファイルを用いて，まずⅢ章のワイヤーフレームモデルを実行し，さらにⅣ章のレイ・トレーシングモデルを，その同じ画面に実行することにより，ワイヤーフレーム上にレイ・トレーシングモデルの画像ができていく様子が分かり，さらにリアルな画像となる（図14）。

図13　レイ・トレーシング法による山

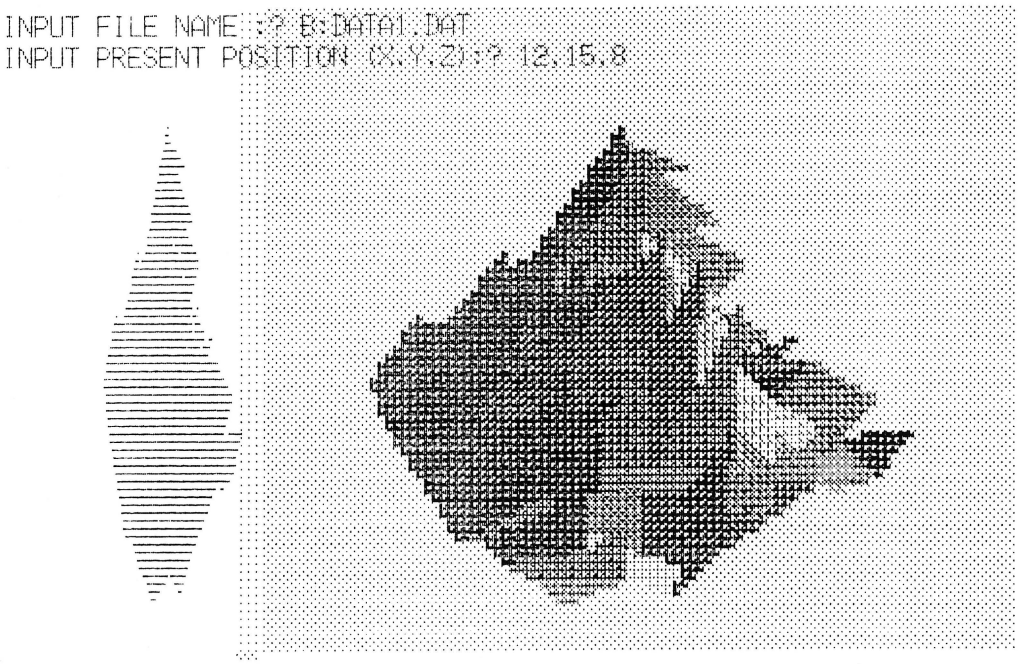
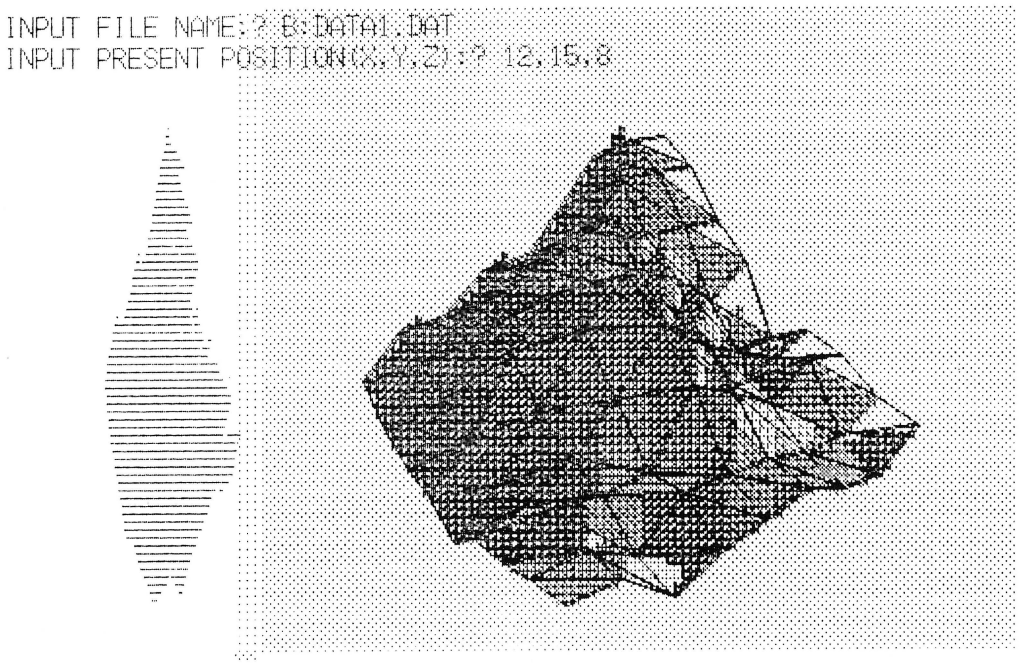


図14　ワイヤーフレーム法とレイ・トレーシング法による山



V　お　わ　り　に

本ノートでは以下のことを明らかにした。

①フラクタル理論に特徴的な「自己相似性」を応用することにより，小量のデータで自然現象を表現することができた。

②3次元図形処理に，コンピュータ・グラフィックスの手法であるワイヤーフレーム・モデルを用いて，比較的簡単に複雑な図形を表現することができた。

③レイ・トレーシング法により，よりリアルな自然現象表現を可能にした。

また，これらによって，人間にとって「らしさ」を簡便に表現できたと考えられる。

自然現象をコンピュータグラフィックスで表現することへの興味は尽きない。そしてそれは「自然現象は人間にとって一体どのように見えるのか」という点と，「自然現象は一体どのようにして形成されているのか」という点に集約されると思う。

今後は，フラクタル理論等のそれぞれの特徴をふまえて，より簡便で，表現力のある方法に発展させていく予定である。

最後に本研究遂行にあたって，有益な御助言を賜わった藤井雅章教授に感謝の意を表します。

付録 PROGRAM 1

```
1000 ' FRACTAL-MOUNTAIN DATA
1010 '
1020 RANDOMIZE VAL(MID$(TIME$,4,2))*60+VAL(RIGHT$(TIME$,2))
1030 WIDTH 80,25
1040 INPUT "N=:";N
1050 SN=2^N
1060 DIM H(SN,SN)
1070 DIM LX(N),UX(N),LY(N),UY(N)
1080 INPUT "INPUT HEIGHT OF CORNER:";H(0,0),H(SN,0),H(SN,SN),H(0,SN)
1090 INPUT "INPUT WID:";WID
1100 COUNT.N=N
1110 LX(N)=0:LY(N)=0:UX(N)=SN:UY(N)=SN
1120 LX=0:LY=0:UX=SN:UY=SN
1130 GOSUB *MID.POINT
1140 INPUT "INPUT DATA FILE NAME:";A$
1150 OPEN A$ FOR OUTPUT AS #1
1160 PRINT #1,SN
1170 FOR I=0 TO SN
1180   FOR J=0 TO SN
1190    PRINT #1,H(I,J)
1200   NEXT J
1210 NEXT I
1220 CLOSE 1
1230 END
1240 *MID.POINT
1280 IF COUNT.N=0 THEN RETURN
1290 H((LX+UX)/2,LY)=WID*(RND-.5)+(H(LX,LY)+H(UX,LY))/2
1300 H((LX+UX)/2,UY)=WID*(RND-.5)+(H(LX,UY)+H(UX,UY))/2
1310 H(LX,(LY+UY)/2)=WID*(RND-.5)+(H(LX,LY)+H(LX,UY))/2
1320 H(UX,(LY+UY)/2)=WID*(RND-.5)+(H(UX,LY)+H(UX,UY))/2
1330 H((LX+UX)/2,(LY+UY)/2)=WID*(RND-.5)+(H(LX,LY)+H(UX,UY)+H(LX,UY)+H(UX,LY))/4
1340 WID=WID/2
1350 UX(COUNT.N-1)=(LX(COUNT.N)+UX(COUNT.N))/2
1360 UY(COUNT.N-1)=(LY(COUNT.N)+UY(COUNT.N))/2
1370 LX(COUNT.N-1)=LX(COUNT.N)
1380 LY(COUNT.N-1)=LY(COUNT.N)
1390 LX=LX(COUNT.N-1):LY=LY(COUNT.N-1):UX=UX(COUNT.N-1):UY=UY(COUNT.N-1)
1400 COUNT.N=COUNT.N-1
1410 GOSUB *MID.POINT
1420 COUNT.N=COUNT.N+1
1430 LX(COUNT.N-1)=UX(COUNT.N-1)
1440 UX(COUNT.N-1)=UX(COUNT.N)
1450 LY(COUNT.N-1)=LY(COUNT.N)
1460 LX=LX(COUNT.N-1):LY=LY(COUNT.N-1):UX=UX(COUNT.N-1):UY=UY(COUNT.N-1)
1470 COUNT.N=COUNT.N-1
1480 GOSUB *MID.POINT
1490 COUNT.N=COUNT.N+1
1500 LY(COUNT.N-1)=UY(COUNT.N-1)
1510 UY(COUNT.N-1)=UY(COUNT.N)
1520 UX(COUNT.N-1)=UX(COUNT.N)
1530 LX=LX(COUNT.N-1):LY=LY(COUNT.N-1):UX=UX(COUNT.N-1):UY=UY(COUNT.N-1)
1540 COUNT.N=COUNT.N-1
1550 GOSUB *MID.POINT
1560 COUNT.N=COUNT.N+1
1570 UX(COUNT.N-1)=LX(COUNT.N-1)
1580 LX(COUNT.N-1)=LX(COUNT.N)
1590 LX=LX(COUNT.N-1):LY=LY(COUNT.N-1):UX=UX(COUNT.N-1):UY=UY(COUNT.N-1)
1600 COUNT.N=COUNT.N-1
1610 GOSUB *MID.POINT
1620 COUNT.N=COUNT.N+1
1630 WID=WID*2
1640 RETURN
```

PROGRAM 2

```
1000 'WAIRE-FRAME
1010 '
1020 SCREEN 3,0,0,1:CLS 2:CONSOLE ,,0,1
1030 INPUT "INPUT FILE NAME:";A$
1040 OPEN A$ FOR INPUT AS #1
1050 INPUT #1,SN
1060 DIM H(SN,SN),IX(SN,SN),IY(SN,SN)
1070 FOR I=0 TO SN
1080   FOR J=0 TO SN
1090    INPUT #1,H(I,J)
1100   NEXT J
1110 NEXT I
1120 PIX%(0,0)=1:PIX%(1,0)=4
1130 PIX%(0,1)=3:PIX%(1,1)=2
1140 INPUT "INPUT PRESENT POSITION(X,Y,Z):";X,Y,Z
1150 IF X=0 THEN HEAD=3.14159/2:GOTO 1170
1160 HEAD=ATN((Y-SN/2)/(X-SN/2))
1170 PITCH=-ATN(Z/SQR((X-SN/2)*(X-SN/2)+(Y-SN/2)*(Y-SN/2)))
1180 GOSUB *TRI
1190 GOSUB *AFFINE
1200 FOR I=0 TO SN
1210   FOR J=0 TO SN-1
1220    LINE(IX(I,J),IY(I,J)-SH.VAL)-(IX(I,J+1),IY(I,J+1)-SH.VAL),4
1230    LINE(IX(I,J)+1,IY(I,J)-SH.VAL+1)-(IX(I,J+1)+1,IY(I,J+1)-SH.VAL+1),4
1240   NEXT J
1250 NEXT I
1260 FOR J=0 TO SN
1270   FOR I=0 TO SN-1
1280    LINE(IX(I,J),IY(I,J)-SH.VAL)-(IX(I+1,J),IY(I+1,J)-SH.VAL),4
1290    LINE(IX(I,J)+1,IY(I,J)-SH.VAL+1)-(IX(I+1,J)+1,IY(I+1,J)-SH.VAL+1),4
1300   NEXT I
1310 NEXT J
1320 FOR J=0 TO SN-1
1330   FOR I=0 TO SN-1
1340    LINE(IX(I+1,J),IY(I+1,J)-SH.VAL)-(IX(I,J+1),IY(I,J+1)-SH.VAL),4
1350    LINE(IX(I+1,J)+1,IY(I+1,J)-SH.VAL+1)-(IX(I,J+1)+1,IY(I,J+1)-SH.VAL+1),4
1360   NEXT I
1370 NEXT J
1380 FOR I=0 TO SN
1390   FOR J=0 TO SN
1400    PSET(IX(I,J),IY(I,J)-SH.VAL),3
1410   NEXT J
1420 NEXT I
1430 END
1440 *TRI
1450 '
1460 '三角関数の計算
1470 '
1480 SH=SIN(HEAD)
1490 CH=COS(HEAD)
1500 SP=SIN(PITCH)
1510 CP=COS(PITCH)
1520 RETURN
1530 *AFFINE
1540 '
1550 'アファイン変換
1560 '
1570 FOR I=0 TO SN
1580   FOR J=0 TO SN
1590    XX=I-X
1600    YY=J-Y
1610    ZZ=H(I,J)-Z
1620    X1=CH*XX+SH*YY
1630    Y1=-SH*XX+CH*YY
1640    Z1=ZZ
1650    X2=CP*X1-SP*Z1
1660    Y2=Y1
1670    Z2=SP*X1+CP*Z1
1680    IX(I,J)=320-Y2/X2*400
1690    IY(I,J)=200+Z2/X2*400
1700   NEXT J
1710 NEXT I
1720 RETURN
```

PROGRAM 3

```
1000 'RAY-TRACING
1010 '
1020 SCREEN 3,0,0,1:CLS 3:CONSOLE ,,0,1
1030 INPUT "INPUT FILE NAME :";A$
1040 OPEN A$ FOR INPUT AS #1
1050 INPUT #1,SN
1060 DIM H(SN,SN),XX(SN,SN),YY(SN,SN),ZZ(SN,SN)
1070 FOR I=0 TO SN
1080   FOR J=0 TO SN
1090    INPUT #1,H(I,J)
1100   NEXT J
1110 NEXT I
1120 INPUT "INPUT PRESENT POSITION (X,Y,Z):";X,Y,Z
1130 IF X<=SN OR Y<=SN THEN PRINT "NOT APPROPRITE POSITION. PLEASE REENTER" :GOTO 1120
1140 HEAD=ATN((Y-SN/2)/(X-SN/2))
1150 PITCH=-ATN(Z/SQR((X-SN/2)*(X-SN/2)+(Y-SN/2)*(Y-SN/2)))
1160 GOSUB *TRI
1170 FOR SY=0 TO 99
1180   IY=SY
1190   FOR SX=30 TO 129
1200    IX=SX
1210    VECT.X=(SX-80)/100
1220    VECT.Y=(50-SY)/100
1230    VX1=-1
1240    VY1=VECT.X
1250    VZ1=VECT.Y
1260    L=SQR(VX1*VX1+VY1*VY1+VZ1*VZ1)
1270    VX1=VX1/L
1280    VY1=VY1/L
1290    VZ1=VZ1/L
1300    VX2=CP*VX1+SP*VZ1
1310    VY2=VY1
1320    VZ2=-SP*VX1+CP*VZ1
1330    VX3=CH*VX2-SH*VY2
1340    VY3=SH*VX2+CH*VY2
1350    VZ3=VZ2
1360    GOSUB *COL.SUR
1370    GOSUB *ANG.SUR
1380    IF NUM.P=0 THEN BLU=80:RED=0:GRN=0:GOSUB *PIXEL:GOTO *SKIP.SX
1390    GRN=PANG1*240+15:RED=PANG2*240+15:BLU=0:GOSUB *PIXEL
1400   *SKIP.SX
1410  NEXT SX
1420 NEXT SY
1430 END
1440 *TRI
1450 '三角関数の計算
1460 '
1470 '
1480 SH=SIN(HEAD)
1490 CH=COS(HEAD)
1500 SP=SIN(PITCH)
1510 CP=COS(PITCH)
1520 RETURN
1530 *COL.SUR
1540 '
1550 '視線と最初に衝突する画をみつける
1560 '
1570 '  OUTPUT :PX%,PY%,UL%,NUM.P,QX,QY,QZ
1580 NUM.P=0
1590 T=(SN-X)/VX3
1600 TOP.Y=INT(Y+T*VY3):IF TOP.Y>=SN THEN TOP.Y=SN-1
1610 IF TOP.Y<0 THEN RETURN
1620 FOR I=SN-1 TO 0 STEP -1
1630   T=(I-X)/VX3
1640   BOT.Y=INT(Y+T*VY3):IF BOT.Y>=SN THEN TOP.Y=SN-1:GOTO *SKIP.Y
1650   IF BOT.Y<0 THEN BOT.Y=0
1660   FOR J=TOP.Y TO BOT.Y STEP -1
1670    UL%=1
1680    GOSUB *SET.P.123
1690    GOSUB *PLANE
1700    V=-(AX*VX3+AY*VY3+AZ*VZ3)
1710    IF V=0 THEN *SKIP.PLANE1
1720    T=(AX*X+AY*Y+AZ*Z+AA)/V
```

```
1730     QX=X+T*VX3:QY=Y+T*VY3:QZ=Z+T*VZ3
1740     IF (QX>I+1) OR (QY>J+1) OR (QX+QY<I+J+1) THEN *SKIP.PLANE1
1750     NUM.P=1:PX%=I:PU%=J:GOTO *END.VP
1760     *SKIP.PLANE1
1770     UL%=0
1780     GOSUB *SET.P.123
1790     GOSUB *PLANE
1800     V=-(AX*VX3+AY*VY3+AZ*VZ3)
1810     IF V=0 THEN *SKIP.PLANE0
1820     T=(AX*X+AY*Y+AZ*Z+AA)/V
1830     QX=X+T*VX3:QY=Y+T*VY3:QZ=Z+T*VZ3
1840     IF (QX<1) OR (QY<J) OR (QX+QY>I+J+1) THEN *SKIP.PLANE0
1850     NUM.P=1:PX%=I:PY%=J:GOTO *END.VP
1860     *SKIP.PLANE0
1870   NEXT J
1880   TOP.Y=BOT.Y
1890 *SKIP.Y:NEXT I
1900 NUM.P=0
1910 *END.VP:RETURN
1920 *ANG.SUR
1930 '
1940 '面と光線の内積を計算
1950 '
1960 ' INPUT :X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3
1970 ' OUTPUT:N.X,N.Y,N.Z
1980 TX1=X2-X1:TY1=Y2-Y1:TZ1=Z2-Z1
1990 TX2=X3-X1:TY2=Y3-Y1:TZ2=Z3-Z1
2000 GOSUB *CAL.N.VECT
2010 PANG1=SQR(1/2)*N.X+SQR(1/2)*N.Z
2020 PANG2=SQR(1/2)*N.Y+SQR(1/2)*N.Z
2030 RETURN
2040 *CAL.N.VECT
2050 '
2060 '法線ベクトル計算
2070 '
2080 '
2090 '
2100 N.X=TY1*TZ2-TZ1*TY2
2110 N.Y=TZ1*TX2-TX1*TZ2
2120 N.Z=TX1*TY2-TY1*TX2
2130 L=SQR(N.X*N.X+N.Y*N.Y+N.Z*N.Z)
2140 IF L=0 THEN RETURN
2150 N.X=N.X/L
2160 N.Y=N.Y/L
2170 N.Z=N.Z/L
2180 RETURN
2190 *SET.P.123
2200 '
2210 '三角形の角の点の計算
2220 '
2230 IF UL%=1 THEN GOTO *UL1
2240 X1=I:Y1=J:Z1=H(I,J)
2250 X2=I+1:Y2=J:Z2=H(I+1,J)
2260 X3=I:Y3=J+1:Z3=H(I,J+1)
2270 RETURN
2280 *UL1
2290 X1=I+1:Y1=J+1:Z1=H(I+1,J+1)
2300 X2=I:Y2=J+1:Z2=H(I,J+1)
2310 X3=I+1:Y3=J:Z3=H(I+1,J)
2320 RETURN
2330 *PLANE
2340 '
2350 '平面の方程式を求める
2360 '
2370 AA=X1*Y2*Z3+Y1*Z2*X3+Z1*X2*Y3-X1*Z2*Y3-Y1*X2*Z3-Z1*Y2*X3
2380 AX=-(Y2*Z3+Y1*Z2+Z1*Y3-Y1*Z3-Z1*Y2-Z2*Y3)
2390 AY=+(X2*Z3+X1*Z2+Z1*X3-Z2*X3-X1*Z3-Z1*X2)
2400 AZ=-(X2*Y3+X1*Y2+Y1*X3-Y2*X3-X1*Y3-Y1*X2)
2410 RETURN
2420 *PIXEL
2430 '
2440 'ピクセルを描く
2450 '
2460 I.BLU=INT(BLU/16)
2470 I.RED=INT(RED/16)
```

```
2480 I.GRN=INT(GRN/16)
2490 IF RND<=(BLU-I.BLU*16)/16 THEN I.BLU=I.BLU+1
2500 IF RND<=(RED-I.RED*16)/16 THEN I.RED=I.RED+1
2510 IF RND<=(GRN-I.GRN*16)/16 THEN I.GRN=I.GRN+1
2520 COL=(I.BLU>=1)+(I.RED>=1)*2+(I.GRN>=1)*4
2530 PSET(IX*4,IY*4),-COL
2540 COL=(I.BLU>=2)+(I.RED>=2)*2+(I.GRN>=2)*4
2550 PSET(IX*4+2,IY*4+2),-COL
2560 COL=(I.BLU>=3)*(I.RED>=3)*2+(I.GRN>=3)*4
2570 PSET(IX*4,IY*4+2),-COL
2580 COL=(I.BLU>=4)*(I.RED>=4)*2+(I.GRN>=4)*4
2590 PSET(IX+4+2,IY*4),-COL
2600 COL=(I.BLU>=5)*(I.RED>=5)*2+(I.GRN>=5)*4
2610 PSET(IX*4+1,IY*4+1),-COL
2620 COL=(I.BLU>=6)*(I.RED>=6)*2+(I.GRN>=6)*4
2630 PSET(IX*4+2+1,IY*4+2+1),-COL
2640 COL=(I.BLU>=7)*(I.RED>=7)*2+(I.GRN>=7)*4
2650 PSET(IX*4+1,IY*4+2+1),-COL
2660 COL=(I.BLU>=8)*(I.RED>=8)*2+(I.GRN>=8)*4
2670 PSET(IX*4+2+1,IY*4+1),-COL
2680 COL=(I.BLU>=9)*(I.RED>=9)*2+(I.GRN>=9)*4
2690 PSET(IX*4+1,IY*4),-COL
2700 COL=(I.BLU>=10)*(I.RED>=10)*2+(I.GRN>=10)*4
2710 PSET(IX*4+2+1,IY*4+2),-COL
2720 COL=(I.BLU>=11)*(I.RED>=11)*2+(I.GRN>=11)*4
2730 PSET(IX*4+1,IY*4+2),-COL
2740 COL=(I.BLU>=12)*(I.RED>=12)*2+(I.GRN>=12)*4
2750 PSET(IX*4+1,IY*4),-COL
2760 COL=(I.BLU>=13)*(I.RED>=13)*2+(I.GRN>=13)*4
2770 PSET(IX*4,IY*4+1),-COL
2780 COL=(I.BLU>=14)*(I.RED>=14)*2+(I.GRN>=14)*4
2790 PSET(IX*4+2,IY*4+2+1),-COL
2800 COL=(I.BLU>=15)+(I.RED>=15)*2+(I.GRN>=15)*4
2810 PSET(IX*4,IY*4+2+1),-COL
2820 COL=(I.BLU>=16)+(I.RED>=16)*2+(I.GRN>=16)*4
2830 PSET(IX*4+2,IY*4+1),-COL
2840 RETURN
```